CLAIMS

What is claimed is:

1. A method for using an allocator to conduct garbage collection, the allocator capable of allocating storage for a user STL object in a shared memory segment common to a plurality of processes, comprising:

creating a first map containing a plurality of nodes representing a plurality of free blocks, each free block represented in the first map by a node in the plurality of nodes denoting a size and an address of the free block;

responding to an allocation request; and

responding to a deletion request by entering a size and an address of a deleted block into the first map.

2. The method of claim 1, further comprising creating a second map containing a plurality of nodes denoting addresses of and pointers to the plurality of free blocks in the first map.

3. The method of claim 1, further comprising coalescing the plurality of free blocks in the first map before entering the size and the address of the deleted block in the first map.

4. The method of claim 1, wherein the responding to an allocation request step comprises returning an address of a free block in the plurality of free blocks if the allocation request is for a storage size equal to or smaller than the free block.

5. The method of claim 1. wherein the responding to an allocation request step comprises allocating memory from the shared memory segment.

6. The method of claim 1, further comprising providing a snapshot of the shared memory segment.

7. The method of claim 6, wherein the snapshot includes a total size of the shared memory segment.

8. The method of claim 1, wherein the first map is available to only one process of the plurality of processes and the one process is a writer process.

9. An apparatus comprising a computer-readable medium having a plurality of sequences of instructions stored thereon including sequences of instructions which, when executed by one or more processors, cause said one or more processors to:

create a first map containing a plurality of nodes representing a plurality of free blocks, each free block represented in the first map by a node in the plurality of nodes denoting a size and an address of the free block;

respond to an allocation request; and

respond to a deletion request by entering a size and an address of a deleted block into the first map.

10. The apparatus of claim 9, the sequences of instructions further comprising instructions to cause said one or more processors to create a second map containing a plurality of nodes denoting addresses of and pointers to the plurality of free blocks in the first map.

11. The apparatus of claim 9, the sequences of instructions further comprising instructions to cause said one more or more processors to coalesce the plurality of free blocks in the first map before entering the size and the address of the deleted block in the first map.

12. The apparatus of claim 9, wherein instructions to respond to the allocation request comprise instructions to return an address of a free block in the plurality of free blocks if the allocation request is for a storage size equal to or smaller than the free block.

13. The apparatus of claim 9, wherein instructions to respond to the allocation request step comprise instructions to allocate memory from the shared memory segment.

14. The apparatus of claim 9, the sequences of instructions further comprising instructions to cause said one or more processors to provide a snapshot of the shared memory segment.

15. The apparatus of claim 14, wherein the snapshot includes a total size of the shared memory segment.

16. The apparatus of claim 9, wherein the first map is available to only one process of the plurality of processes and the one process is a writer process.

17. A computer-implemented method for sharing data structures among processes by using a shared memory segment, comprising:

creating the shared memory segment;

anchoring a system STL (Standard Template Library) map in the shared memory segment;

receiving a user STL object;

obtaining an address for the user STL object; and

inserting the user STL object into the system STL map.

18. The method of claim 17, wherein the shared memory segment is named and comprises approximately l0MB of memory.

19. The method of claim 17, wherein receiving the user STL object comprises creating the user STL object.

20. The method of claim 17, wherein the user STL object has a name.

21. The method of claim 20, wherein the obtaining step comprises returning the address of the user STL object if the name of the user STL object is in the system STL map.

22. The method of claim 20, wherein the obtaining step comprises returning a next allocation address from the shared memory segment if the name of the user STL object is not in the system STL map, and allocating space for the user STL object.

23. The method of claim 22, further comprising updating the next allocation address by increasing the next allocation address by an allocated size of the user STL object.

24. The method of claim 23, wherein an access to the nest allocation address is synchronized.

25. The method of claim 17, wherein the inserting step comprises entering the name and the address of the user STL object into the system STL map.

26. The method of claim 17, further comprising requesting storage from an allocator, wherein the allocator allocates memory from the shared memory segment to store at least one node of the user STL object and data added to the user STL object.

27. The method of claim 26, wherein the data comprise an entity capable of supporting a writer and a plurality of readers.

28. The method of claim 26, wherein the allocator performs garbage collection.

29. The method of claim 1, wherein the processes comprise threaded processes.